Simulation

Computer Simulation

- Computer simulations are experiments performed on the computer using computer-generated random numbers.
- Simulation is used to
  - study the behavior of complex systems such as
    - biological systems
    - ecosystems
    - engineering systems
    - computer networks
  - compute values of otherwise intractable quantities such as integrals
  - maximize or minimize the value of a complicated function
  - study the behavior of statistical procedures
  - implement novel methods of statistical inference
- Simulations need
  - uniform random numbers
  - non-uniform random numbers
  - random vectors, stochastic processes, etc.
  - techniques to design good simulations
  - methods to analyze simulation results

Uniform Random Numbers

- The most basic distribution is the uniform distribution on $[0,1]$
- Ideally we would like to be able to obtain a sequence of independent draws from the uniform distribution on $[0,1]$.
- Since we can only use finitely many digits, we can also work with
  - A sequence of independent discrete uniform random numbers on $\{0,1,\ldots,M-1\}$ or $\{1,2,\ldots,M\}$ for some large M.
  - A sequence of independent random bits with equal probability for 0 and 1.
- Some methods are based on physical processes such as
  - nuclear decay

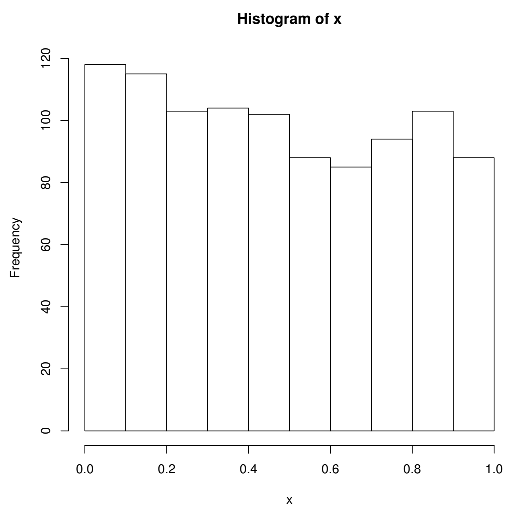
    `http://www.fourmilab.ch/hotbits/`
  - atmospheric noise

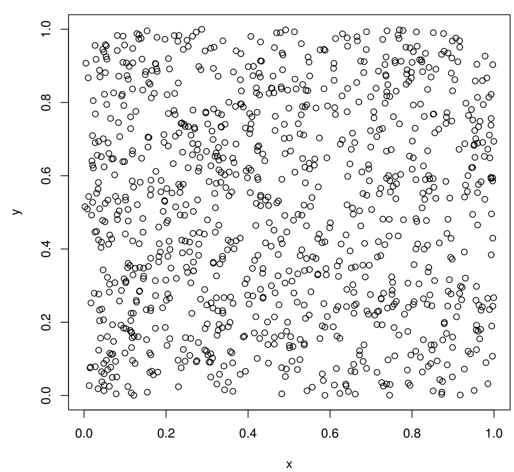
    `http://www.random.org/`

    The R package random provides an interface.
  - air turbulence over disk drives or thermal noise in a semiconductor (Toshiba Random Master PCI device)
  - event timings in a computer (Linux `/dev/random`)

Using **/dev/random** from R

```
devRand <- file("/dev/random", open="rb")
U <- function()
    (as.double(readBin(devRand, "integer"))+2^31) / 2^32
x <-numeric(1000)
for (i in seq(along=x)) x[i] <- U()
hist(x)
y <- numeric(1000)
for (i in seq(along=x)) y[i] <- U()
plot(x,y)
close(devRand)
```





Issues with Physical Generators

- can be very slow
- not reproducible except by storing all values
- distribution is usually not exactly uniform; can be off by enough to matter
- departures from independence may be large enough to matter
- mechanisms, defects, are hard to study
- can be improved by combining with other methods

Pseudo-Random Numbers

Pseudo-random number generators produce a sequence of numbers that is

- not random
- easily reproducible
- "unpredictable;" "looks random"
- behaves in many respects like a sequence of independent draws from a (discretized) uniform $[0,1]$ distribution
- fast to produce

Pseudo-random generators come in various qualities

- Simple generators
  - easy to implement
  - run very fast
  - easy to study theoretically
  - usually have known, well understood flaws
- More complex
  - often based on combining simpler ones
  - somewhat slower but still very fast
  - sometimes possible to study theoretically, often not
  - guaranteed to have flaws; flaws may not be well understood (yet)
- Cryptographic strength

  `https://www.schneier.com/fortuna.html`

  - often much slower, more complex
  - thought to be of higher quality
  - may have legal complications
  - weak generators can enable exploits, a recent issue in iOS 7

We use mostly generators in the first two categories.

General Properties

- Most pseudo-random number generators produce a sequence of integers $x_1, x_2, \ldots$ in the range $\{0, 1, \ldots, M-1\}$ for some M using a recursion of the form
$$x_n = f(x_{n-1}, x_{n-2}, \ldots, x_{n-k})$$
- Values $u_1, u_2, \ldots$ are then produced by
$$u_i = g(x_{di}, x_{di-1}, \ldots, x_{di-d+1})$$
- Common choices of M are
  - $M = 2^{31}$ or $M = 2^{32}$
  - $M = 2^{31} - 1$, a *Mersenne prime*
  - $M = 2$ for bit generators
- The value k is the *order* of the generator
- The set of the most recent k values is the *state* of the generator.
- The initial state $x_1, \ldots, x_k$ is called the *seed*.
- Since there are only finitely many possible states, eventually these generators will repeat.
- The length of a cycle is called the *period* of a generator.
- The maximal possible period is on the order of M^k
- Needs change:
  - As computers get faster, larger, more complex simulations are run.
  - A generator with period 2^{32} used to be good enough.
  - A current computer can run through 2^{32} pseudo-random numbers in under one minute.
  - Most generators in current use have periods 2^{64} or more.
  - Parallel computation also raises new issues.

Linear Congruential Generators

- A linear congruential generator is of the form

$$x_i = (ax_{i-1} + c) \mod M$$

  with $0 \leq x_i < M$.

  - a is the *multiplier*
  - c is the *increment*
  - M is the *modulus*

- A multiplicative generator is of the form

$$x_i = ax_{i-1} \mod M$$

  with $0 < x_i < M$.

- A linear congruential generator has full period M if and only if three conditions hold:

  - $\gcd(c, M) = 1$
  - $a \equiv 1 \mod p$ for each prime factor p of M
  - $a \equiv 1 \mod 4$ if 4 divides M

- A multiplicative generator has period at most $M - 1$. Full period is achieved if and only if M is prime and a is a *primitive root modulo* M, i.e. $a \neq 0$ and $a^{(M-1)/p} \not\equiv 1 \mod M$ for each prime factor p of $M - 1$.

Examples

- Lewis, Goodman, and Miller ("minimal standard" of Park and Miller):

  $$x_i = 16807 x_{i-1} \mod (2^{31}-1) = 7^5 x_{i-1} \mod (2^{31}-1)$$

  Reasonable properties, period $2^{31}-2 \approx 2.15 * 10^9$ is very short for modern computers.

- RANDU:

  $$x_i = 65538 x_{i-1} \mod 2^{31}$$

  Period is only 2^{29} but that is the least of its problems:

  $$u_{i+2} - 6u_{i+1} + 9u_i = \text{an integer}$$

  so (u_i, u_{i+1}, u_{i+2}) fall on 15 parallel planes. Using the `randu` data set and the `rgl` package:

  ```
  library(rgl)
  points3d(randu)
  par3d(FOV=1)  ## removes perspective distortion
  ```

  With a larger number of points:

  ```
  seed <- as.double(1)
  RANDU <- function() {
      seed <<- ((2^16 + 3) * seed) %% (2^31)
      seed/(2^31)
  }

  U <- matrix(replicate(10000 * 3, RANDU()), ncol = 3, byrow = TRUE)
  clear3d()
  points3d(U)
  par3d(FOV=1)
  ```

  This generator used to be the default generator on IBM 360/370 and DEC PDP11 machines.

Some examples are available in

```
http://www.stat.uiowa.edu/~luke/classes/STAT7400/
                examples/sim.Rmd
```

Lattice Structure

- All linear congruential sequences have a *lattice structure*
- Methods are available for computing characteristics, such as maximal distance between adjacent parallel planes
- Values of M and a can be chosen to achieve good lattice structure for $c = 0$ or $c = 1$; other values of c are not particularly useful.

Shift-Register Generators

- Shift-register generators take the form

$$x_i = a_1 x_{i-1} + a_2 x_{i-2} + \cdots + a_p x_{i-p} \mod 2$$

  for binary constants $a_1, \dots, a_p$.

- values in $[0,1]$ are often constructed as

$$u_i = \sum_{s=1}^{L} 2^{-s} x_{ti+s} = 0.x_{it+1} x_{it+2} \dots x_{it+L}$$

  for some t and $L \leq t$. t is the *decimation*.

- The maximal possible period is $2^p - 1$ since all zeros must be excluded.

- The maximal period is achieved if and only if the polynomial

$$z^p + a_1 z^{p-1} + \cdots + a_{p-1} z + a_p$$

  is irreducible over the finite field of size 2.

- Theoretical analysis is based on k-distribution: A sequence of M bit integers with period $2^p - 1$ is *k-distributed* if every k-tuple of integers appears 2^{p-kM} times, except for the zero tuple, which appears one time fewer.

- Generators are available that have high periods and good k-distribution properties.

Lagged Fibonacci Generators

- Lagged Fibonacci generators are of the form

$$x_i = (x_{i-k} \circ x_{i-j}) \mod M$$

for some binary operator $\circ$.

- Knuth recommends

$$x_i = (x_{i-100} - x_{i-37}) \mod 2^{30}$$

  - There are some regularities if the full sequence is used; one recommendation is to generate in batches of 1009 and use only the first 100 in each batch.
  - Initialization requires some care.

Combined Generators

- Combining several generators may produce a new generator with better properties.
- Combining generators can also fail miserably.
- Theoretical properties are often hard to develop.
- Wichmann-Hill generator:

$$\begin{aligned} x_i &= 171 x_{i-1} \mod 30269 \\ y_i &= 172 y_{i-1} \mod 30307 \\ z_i &= 170 z_{i-1} \mod 30323 \end{aligned}$$

and

$$u_i = \left(\frac{x_i}{30269} + \frac{y_i}{30307} + \frac{z_i}{30232}\right) \mod 1$$

The period is around 10^{12}.

This turns out to be equivalent to a multiplicative generator with modulus

$$M = 27817185604309$$

- Marsaglia's Super-Duper used in S-PLUS and others combines a linear congruential and a feedback-shift generator.

Other Generators

- Mersenne twister
- Marsaglia multicarry
- Parallel generators
  - SPRNG `http://sprng.cs.fsu.edu`.
  - L'Ecuyer, Simard, Chen, and Kelton

    `http://www.iro.umontreal.ca/~lecuyer/myftp/streams00/`

Pseudo-Random Number Generators in R

R provides a number of different basic generators:

Wichmann-Hill: Period around 10^{12}

Marsaglia-Multicarry: Period at least 10^{18}

Super-Duper: Period around 10^{18} for most seeds; similar to S-PLUS

Mersenne-Twister: Period $2^{19937}-1 \approx 10^{6000}$; equidistributed in 623 dimensions; current default in R.

Knuth-TAOCP: Version from second edition of *The Art of Computer Programming, Vol. 2*; period around 10^{38}.

Knuth-TAOCP-2002: From third edition; differs in initialization.

L'Ecuyer-CMRG: A combined multiple-recursive generator from L'Ecuyer (1999). The period is around 2^{191}. This provides the basis for the multiple streams used in package `parallel`.

user-supplied: Provides a mechanism for installing your own generator; used for parallel generation by

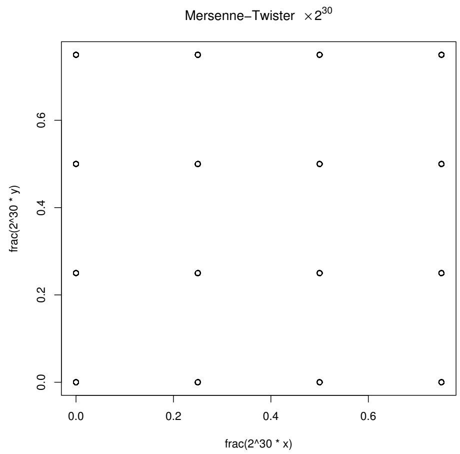
- `rsprng` package interface to SPRNG
- `rlecuyer` package interface to L'Ecuyer, Simard, Chen, and Kelton system
- `rstreams` package, another interface to L'Ecuyer et al.

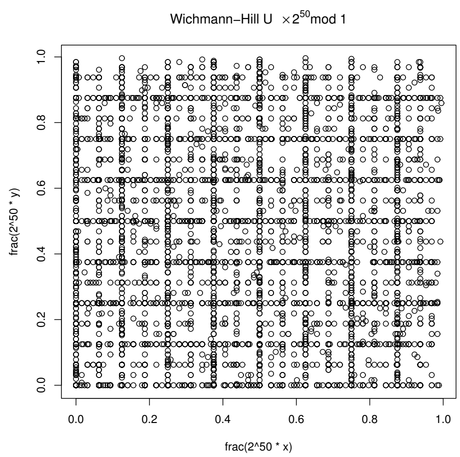
Testing Generators

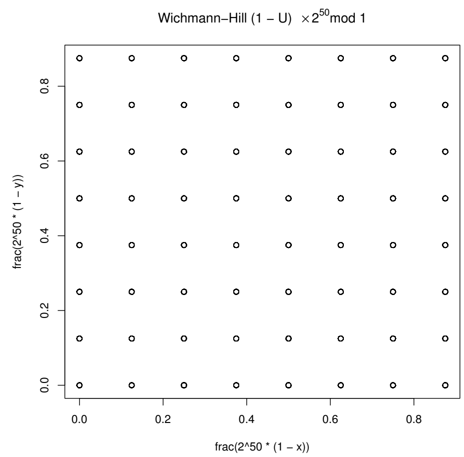
- All generators have flaws; some are known, some are not (yet).
- Tests need to look for flaws that are likely to be important in realistic statistical applications.
- Theoretical tests look for
  - bad lattice structure
  - lack of k-distribution
  - other tractable properties
- Statistical tests look for simple simulations where pseudo-random number streams produce results unreasonably far from known answers.
- Some batteries of tests:
  - DIEHARD `http://stat.fsu.edu/pub/diehard/`
  - DIEHARDER `http://www.phy.duke.edu/~rgb/General/dieharder.php`
  - NIST Test Suite `http://csrc.nist.gov/groups/ST/toolkit/rng/`
  - TestU01 `http://www.iro.umontreal.ca/~lecuyer`

Issues and Recommendations

- Good choices of generators change with time and technology.
  - Faster computers need longer periods.
  - Parallel computers need different methods.
- All generators are flawed
  - Bad simulation results due to poor random number generators are very rare; coding errors in simulations are not.
  - Testing a generator on a "similar" problem with known answers is a good idea (and may be useful to make results more accurate).
  - Using multiple generators is a good idea; R makes this easy to do.
  - Be aware that some generators can produce uniforms equal to 0 or 1 (I believe R's will not).
  - Avoid methods that are sensitive to low order bits


Mersenne−Twister $\times 2^{30}$
frac(2^30 * y)
frac(2^30 * x)


Wichmann−Hill U $\times 2^{50}$mod 1
frac(2^50 * y)
frac(2^50 * x)


Wichmann−Hill (1 − U) $\times 2^{50}$mod 1
frac(2^50 * (1 − y))
frac(2^50 * (1 − x))

Non-Uniform Random Variate Generation

- Starting point: Assume we can generate a sequence of independent uniform $[0,1]$ random variables.
- Develop methods that generate general random variables from uniform ones.
- Considerations:
  - Simplicity, correctness
  - Accuracy, numerical issues
  - Speed
    - Setup
    - Generation
- General approaches:
  - Univariate transformations
  - Multivariate transformations
  - Mixtures
  - Accept/Reject methods

Inverse CDF Method

Suppose F is a cumulative distribution function (CDF). Define the inverse CDF as

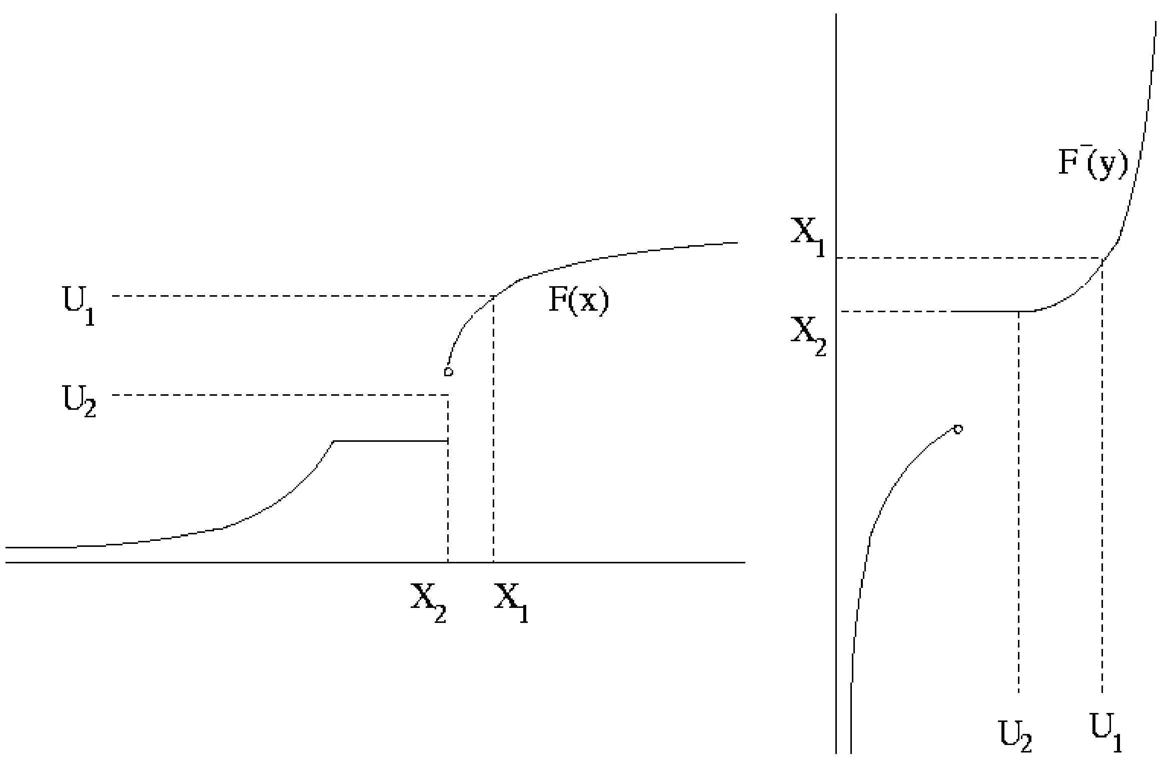
$$F^{-}(u) = \min\{x : F(x) \geq u\}$$

If $U \sim \mathrm{U}[0,1]$ then $X = F^{-}(U)$ has CDF F.

Proof. Since F is right continuous, the minimum is attained. Therefore $F(F^{-}(u)) \geq u$ and $F^{-}(F(x)) = \min\{y : F(y) \geq F(x)\}$. So

$$\{(u,x) : F^{-}(u) \leq x\} = \{(u,x) : u \leq F(x)\}$$

and thus $P(X \leq x) = P(F^{-}(U) \leq x) = P(U \leq F(x)) = F(x)$. $\square$



Example: Unit Exponential Distribution

The unit exponential CDF is

$$F(x) = \begin{cases} 1 - e^{-x} & \text{for } x > 0 \\ 0 & \text{otherwise} \end{cases}$$

with inverse CDF

$$F^{-}(u) = -\log(1-u)$$

So $X = -\log(1-U)$ has an exponential distribution.

Since $1 - U \sim \mathrm{U}[0,1]$, $-\log U$ also has a unit exponential distribution.

If the uniform generator can produce 0, then these should be rejected.

Example: Standard Cauchy Distribution

The CDF of the standard Cauchy distribution is

$$F(x) = \frac{1}{2} + \frac{1}{\pi}\arctan(x)$$

with inverse CDF

$$F^{-}(u) = \tan(\pi(u - 1/2))$$

So $X = \tan(\pi(U - 1/2))$ has a standard Cauchy distribution.

An alternative form is: Let U_1, U_2 be independent $\mathrm{U}[0,1]$ random variables and set

$$X = \begin{cases} \tan(\pi(U_2/2) & \text{if } U_1 \geq 1/2 \\ -\tan(\pi(U_2/2) & \text{if } U_1 < 1/2 \end{cases}$$

- U_1 produces a random sign
- U_2 produces the magnitude
- This will preserve fine structure of U_2 near zero, if there is any.

Example: Standard Normal Distribution

The CDF of the standard normal distribution is

$$\Phi(x) = \int_{-\infty}^{x} \frac{1}{\sqrt{2\pi}} e^{-z^2/2} dz$$

and the inverse CDF is Φ^{-1}.

- Neither Φ nor Φ^{-1} are available in closed form.
- Excellent numerical routines are available for both.
- Inversion is currently the default method for generating standard normals in R.
- The inversion approach uses two uniforms to generate one higher-precision uniform via the code

```
    case INVERSION:
#define BIG 134217728 /* 2^27 */
        /* unif_rand() alone is not of high enough precision */
        u1 = unif_rand();
        u1 = (int)(BIG*u1) + unif_rand();
        return qnorm5(u1/BIG, 0.0, 1.0, 1, 0);
```

Example: Geometric Distribution

The geometric distribution with PMF $f(x) = p(1-p)^x$ for $x = 0, 1, \ldots,$ has CDF

$$F(x) = \begin{cases} 1-(1-p)^{\lfloor x+1 \rfloor} & \text{for } x \geq 0 \\ 0 & \text{for } x < 0 \end{cases}$$

where $\lfloor y \rfloor$ is the integer part of y. The inverse CDF is

$$\begin{aligned} F^{-}(u) &= \lceil \log(1-u)/\log(1-p) \rceil - 1 \\ &= \lfloor \log(1-u)/\log(1-p) \rfloor \qquad \text{except at the jumps} \end{aligned}$$

for $0 < u < 1$. So $X = \lfloor \log(1-U)/\log(1-p) \rfloor$ has a geometric distribution with success probability p.

Other possibilities:

$$X = \lfloor \log(U)/\log(1-p) \rfloor$$

or

$$X = \lfloor -Y/\log(1-p) \rfloor$$

where Y is a unit exponential random variable.

Example: Truncated Normal Distribution

Suppose $X \sim \mathrm{N}(\mu, 1)$ and

$$y \sim X | X > 0.$$

The CDF of Y is

$$F_Y(y) = \begin{cases} \frac{P(0<X\leq y)}{P(0<X)} & \text{for } y \geq 0 \\ 0 & \text{for } y < 0 \end{cases} = \begin{cases} \frac{F_X(y)-F_X(0)}{1-F_X(0)} & \text{for } y \geq 0 \\ 0 & \text{for } y < 0 \end{cases}.$$

The inverse CDF is

$$F_Y^{-1}(u) = F_X^{-1}(u(1-F_X(0)) + F_X(0)) = F_X^{-1}(u + (1-u)F_X(0)).$$

An R function corresponding to this definition is

```
Q1 <- function(p, m) qnorm(p + (1 - p)* pnorm(0, m), m)
```

This seems to work well for positive μ but not for negative values far from zero:

```
> Q1(0.5, c(1, 3, 5, 10, -10))
[1]  1.200174  3.001692  5.000000 10.000000       Inf
```

The reason is that `pnorm(0, -10)` is rounded to one.

A mathematically equivalent formulation of the inverse CDF is

$$F_Y^{-1}(u) = F_X^{-1}(1-(1-u)(1-F_X(0)))$$

which leads to

```
Q2 <- function(p, m)
    qnorm((1 - p)* pnorm(0, m, lower.tail = FALSE),
          m, lower.tail = FALSE)
```

and

```
> Q2(0.5, c(1, 3, 5, 10, -10))
[1]  1.20017369  3.00169185  5.00000036 10.00000000  0.06841184
```

Issues

- In principle, inversion can be used for any distribution.
- Sometimes routines are available for F^{-} but are quite expensive:

```
> system.time(rbeta(1000000, 2.5, 3.5))
   user  system elapsed
  0.206   0.000   0.211
> system.time(qbeta(runif(1000000), 2.5, 3.5))
   user  system elapsed
  4.139   0.001   4.212
```

  `rbeta` is about 20 times faster than inversion.
- If F^{-} is not available but F is, then one can solve the equation $u = F(x)$ numerically for x.
- Accuracy of F or F^{-} may be an issue, especially when writing code for a parametric family that is to work well over a wide parameter range.
- Even when inversion is costly,
  - the cost of random variate generation may be a small fraction of the total cost of a simulation
  - using inversion creates a simple relation between the variables and the underlying uniforms that may be useful

Multivariate Transformations

Many distributions can be expressed as the marginal distribution of a function of several variables.

Box-Muller Method for the Standard Normal Distribution

Suppose X_1 and X_2 are independent standard normals. The polar coordinates θ and R are independent,

- θ is uniform on $[0, 2\pi)$
- R^2 is χ^2_2, which is exponential with mean 2

So if U_1 and U_2 are independent and uniform on $[0, 1]$, then

$$\begin{aligned} X_1 &= \sqrt{-2 \log U_1} \cos(2\pi U_2) \\ X_2 &= \sqrt{-2 \log U_1} \sin(2\pi U_2) \end{aligned}$$

are independent standard normals. This is the *Box-Muller method*.

Polar Method for the Standard Normal Distribution

The trigonometric functions are somewhat slow to compute. Suppose (V_1, V_2) is uniform on the unit disk

$$\{(v_1, v_2) : v_1^2 + v_2^2 \leq 1\}$$

Let $R^2 = V_1^2 + V_2^2$ and

$$\begin{aligned} X_1 &= V_1 \sqrt{-(2 \log R^2)/R^2} \\ X_2 &= V_2 \sqrt{-(2 \log R^2)/R^2} \end{aligned}$$

Then X_1, X_2 are independent standard normals.

This is the *polar method* of Marsaglia and Bray.

Generating points uniformly on the unit disk can be done using *rejection sampling*, or *accept/reject sampling:*

repeat
  generate independent $V_1, V_2 \sim \mathrm{U}(-1,1)$
until $V_1^2 + V_2^2 \leq 1$
return (V_1, V_2)

- This independently generates pairs (V_1, V_2) uniformly on the square $(-1,1) \times (-1,1)$ until the result is inside the unit disk.
- The resulting pair is uniformly distributed on the unit disk.
- The number of pairs that need to be generated is a geometric variable with success probability

$$p = \frac{\text{area of disk}}{\text{area of square}} = \frac{\pi}{4}$$

  The expected number of generations needed is $1/p = 4/\pi = 1.2732$.
- The number of generations needed is independent of the final pair.

Polar Method for the Standard Cauchy Distribution

The ratio of two standard normals has a Cauchy distribution.

Suppose two standard normals are generated by the polar method,

$$X_1 = V_1\sqrt{-(2\log R^2)/R^2}$$
$$X_2 = V_2\sqrt{-(2\log R^2)/R^2}$$

with $R^2 = V_1^2 + V_2^2$ and (V_1, V_2) uniform on the unit disk. Then

$$Y = \frac{X_1}{X_2} = \frac{V_1}{V_2}$$

is the ratio of the two coordinates of the pair that is uniformly distributed on the unit disk.

This idea leads to a general method, the *Ratio-of-Uniforms method*.

Student's t Distribution

Suppose

- Z has a standard normal distribution,
- Y has a χ^2_ν distribution,
- Z and Y are independent.

Then

$$X = \frac{Z}{\sqrt{Y/\nu}}$$

has a t distribution with ν degrees of freedom.

To use this representation we will need to be able to generate from a χ^2_ν distribution, which is a Gamma$(\nu/2, 2)$ distribution.

Beta Distribution

Suppose $\alpha > 0$, $\beta > 0$, and

- $U \sim \text{Gamma}(\alpha, 1)$
- $V \sim \text{Gamma}(\beta, 1)$
- U, V are independent

Then

$$X = \frac{U}{U+V}$$

has a Beta(α, β) distribution.

F Distribution

Suppose $a > 0$, $b > 0$, and

- $U \sim \chi^2_a$
- $V \sim \chi^2_b$
- U, V are independent

Then

$$X = \frac{U/a}{V/b}$$

has an F distribution with a and b degrees of freedom.

Alternatively, if $Y \sim \text{Beta}(a/2, b/2)$, then

$$X = \frac{b}{a}\frac{Y}{1-Y}$$

has an F distribution with a and b degrees of freedom.

Non-Central t Distribution

Suppose

- $Z \sim \mathrm{N}(\mu, 1)$,
- $Y \sim \chi^2_\nu$,
- Z and Y are independent.

Then

$$X = \frac{Z}{\sqrt{Y/\nu}}$$

has non-central t distribution with ν degrees of freedom and non-centrality parameter μ.

Non-Central Chi-Square, and F Distributions

Suppose

- $Z_1, \ldots, Z_k$ are independent
- $Z_i \sim \mathrm{N}(\mu_i, 1)$

Then
$$Y = Z_1^2 + \cdots + Z_k^2$$
has a non-central chi-square distribution with k degrees of freedom and non-centrality parameter
$$\delta = \mu_1^2 + \cdots + \mu_k^2$$
An alternative characterization: if $\widetilde{Z}_1, \ldots, \widetilde{Z}_k$ are independent standard normals then
$$Y = (\widetilde{Z}_1 + \sqrt{\delta})^2 + \widetilde{Z}_2^2 \cdots + \widetilde{Z}_k^2 = (\widetilde{Z}_1 + \sqrt{\delta})^2 + \sum_{i=2}^{k} \widetilde{Z}_i^2$$
has a non-central chi-square distribution with k degrees of freedom and non-centrality parameter δ.

The non-central F is of the form
$$X = \frac{U/a}{V/b}$$
where U, V are independent, U is a non-central χ_a^2 and V is a central χ_b^2 random variable.

Bernoulli and Binomial Distributions

Suppose $p \in [0,1]$, $U \sim \mathrm{U}[0,1]$, and

$$X = \begin{cases} 1 & \text{if } U \leq p \\ 0 & \text{otherwise} \end{cases}$$

Then X bas a Bernoulli(p) distribution.

If $Y_1, \ldots, Y_n$ are independent Bernoulli(p) random variables, then

$$X = Y_1 + \cdots + Y_n$$

has a Binomial(n, p) distribution.

For small n this is an effective way to generate binomials.

Mixtures and Conditioning

Many distributions can be expressed using a hierarchical structure:

$$\begin{aligned} X|Y &\sim f_{X|Y}(x|y) \\ Y &\sim f_Y(y) \end{aligned}$$

The marginal distribution of X is called a *mixture distribution*. We can generate X by

Generate Y from $f_Y(y)$.
Generate $X|Y = y$ from $f_{X|Y}(x,y)$.

Student's t Distribution

Another way to think of the t_ν distribution is:

$$\begin{aligned} X|Y &\sim \mathrm{N}(0, \nu/Y) \\ Y &\sim \chi^2_\nu \end{aligned}$$

The t distribution is a *scale mixture of normals*.

Other choices of the distribution of Y lead to other distributions for X.

Negative Binomial Distribution

The negative binomial distribution with PMF

$$f(x) = \binom{x+r-1}{r-1} p^r (1-p)^x$$

for $x = 0, 1, 2, \ldots$, can be written as a gamma mixture of Poissons: if

$$\begin{aligned} X|Y &\sim \text{Poisson}(Y) \\ Y &\sim \text{Gamma}(r, (1-p)/p) \end{aligned}$$

then $X \sim$ Negative Binomial(r, p).

[The notation Gamma(α, β) means that β is the scale parameter.]

This representation makes sense even when r is not an integer.

Non-Central Chi-Square

The density of the non-central χ^2_ν distribution with non-centrality parameter δ is

$$f(x) = e^{-\delta/2} \sum_{i=0}^{\infty} \frac{(\delta/2)^i}{i!} f_{\nu+2i}(x)$$

where $f_k(x)$ central χ^2_k density. This is a Poisson-weighted average of χ^2 densities, so if

$$\begin{aligned} X|Y &\sim \chi^2_{\nu+2Y} \\ Y &\sim \text{Poisson}(\delta/2) \end{aligned}$$

then X has a non-central χ^2_ν distribution with non-centrality parameter δ.

Composition Method

Suppose we want to sample from the density

$$f(x) = \begin{cases} x/2 & 0 \le x < 1 \\ 1/2 & 1 \le x < 2 \\ 3/2 - x/2 & 2 \le x \le 3 \\ 0 & \text{otherwise} \end{cases}$$

We can write f as the mixture

$$f(x) = \frac{1}{4} f_1(x) + \frac{1}{2} f_2(x) + \frac{1}{4} f_3(x)$$

with

$$\begin{aligned} f_1(x) &= 2x & 0 \le x < 1 \\ f_2(x) &= 1 & 1 \le x < 2 \\ f_3(x) &= 6 - 2x & 2 \le x \le 3 \end{aligned}$$

and $f_i(x) = 0$ for other values of x.

Generating from the f_i is straight forward. So we can sample from f using:

Generate I from $\{1,2,3\}$ with probabilities $1/4, 1/2, 1/4$.
Generate X from $f_I(x)$ by inversion.

This approach can be used in conjunction with other methods.

One example: The polar method requires sampling uniformly from the unit disk. This can be done by

- encloseing the unit disk in a regular hexagon
- using composition to sample uniformly from the hexagon until the result is in the unit disk.

Alias Method

Suppose $f(x)$ is a probability mass function on $\{1,2,\ldots,k\}$. Then $f(x)$ can be written as

$$f(x) = \sum_{i=1}^{k} \frac{1}{k} f_i(x)$$

where

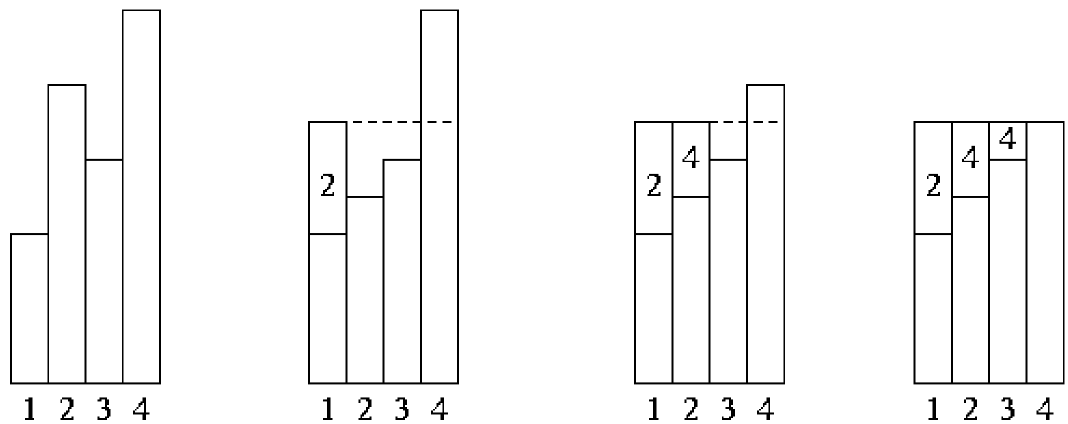
$$f_i(x) = \begin{cases} q_i & x = i \\ 1-q_i & x = a_i \\ 0 & \text{otherwise} \end{cases}$$

for some $q_i \in [0,1]$ and some $a_i \in \{1,2,\ldots,k\}$.

Once values for q_i and a_i have been found, generation is easy:

> Generate I uniform on $\{1,\ldots,k\}$
> Generate U uniform on $[0,1]$
> **if** $U \leq q_I$
>   **return** I
> **else**
>   **return** a_I

The setup process used to compute the q_i and a_i is called *leveling the histogram*:



This is *Walker's alias method*.

A complete description is in Ripley (1987, Alg 3.13B).

The alias method is an example of trading off a setup cost for fast generation.

The alias method is used by the `sample` function for unequal probability sampling with replacement when there are enough reasonably probable values.

`https://svn.r-project.org/R/trunk/src/main/random.c`

Accept/Reject Methods

Sampling Uniformly from the Area Under a Density

Suppose h is a function such that

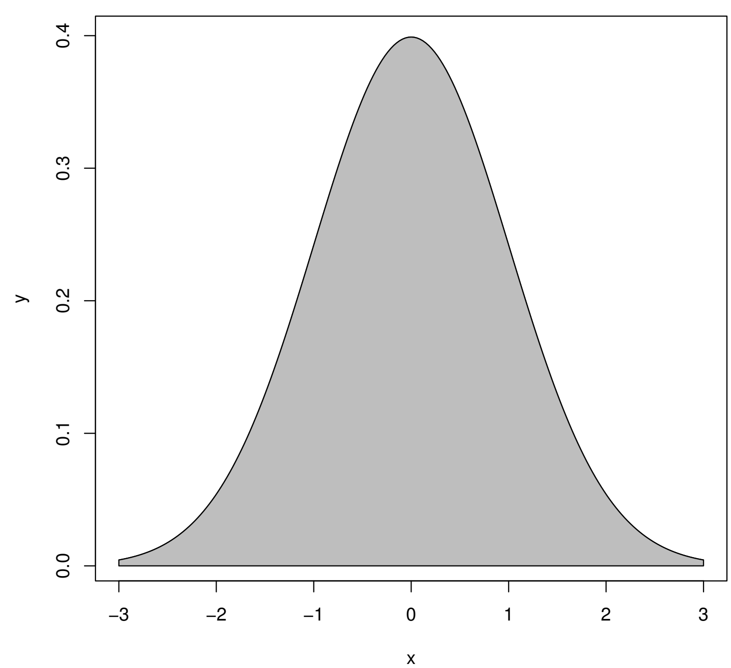
- $h(x) \geq 0$ for all x
- $\int h(x)dx < \infty$.

Let

$$\mathscr{G}_h = \{(x,y) : 0 < y \leq h(x)\}$$

The area of $\mathscr{G}_h$ is

$$|\mathscr{G}_h| = \int h(x)dx < \infty$$



Suppose (X,Y) is uniformly distributed on $\mathscr{G}_h$. Then

- The conditional distribution of $Y|X=x$ is uniform on $(0,h(x))$.

- The marginal distribution of X has density $f_X(x) = h(x)/\int h(y)dy$:

$$f_X(x) = \int_0^{h(x)} \frac{1}{|\mathscr{G}_h|} dy = \frac{h(x)}{\int h(y)dy}$$

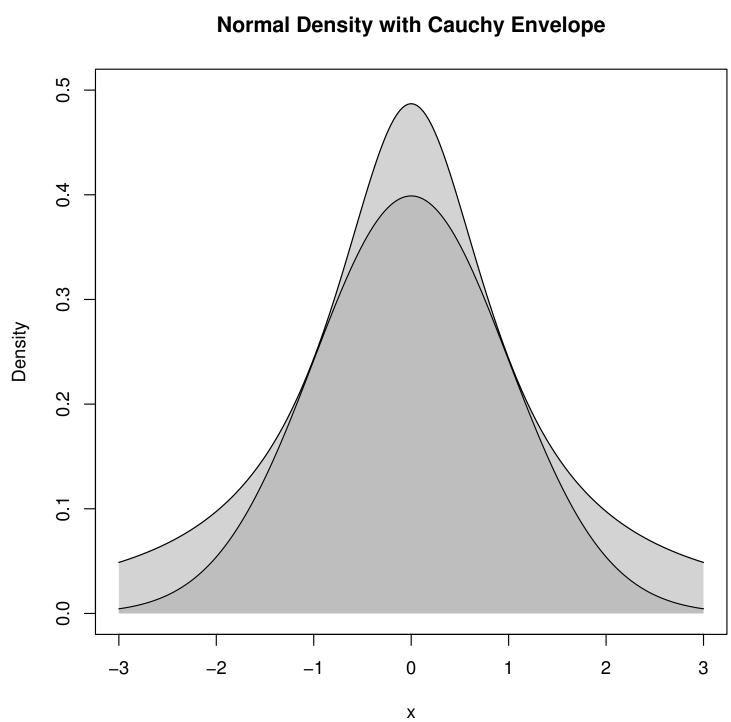
Rejection Sampling Using an Envelope Density

Suppose g is a density and $M > 0$ is a real number such that

$$h(x) \leq Mg(x) \quad \text{for all } x$$

or, equivalently,

$$\sup \frac{h(x)}{g(x)} \leq M \quad \text{for all } x$$



$Mg(x)$ is an *envelope* for $h(x)$.

Suppose

- we want to sample from a density proportional to h
- we can find a density g and a constant M such that
  - $Mg(x)$ is an envelope for $h(x)$
  - it is easy to sample from g

Then

- we can sample X from g and $Y|X = x$ from $\mathrm{U}(0, Mg(x))$ to get a pair (X,Y) uniformly distributed on $\mathscr{G}_{Mg}$
- we can repeat until the pair (X,Y) satisfies $Y \leq h(X)$
- the resulting pair (X,Y) is uniformly distributed on $\mathscr{G}_h$
- so the marginal density of the resulting X is $f_X(x) = h(x)/\int h(y)dy$.
- the number of draws from the uniform distribution on $\mathscr{G}_{Mg}$ needed until we obtain a pair in $\mathscr{G}_h$ is independent of the final pair
- the number of draws has a geometric distribution with success probability
  $$p = \frac{|\mathscr{G}_h|}{|\mathscr{G}_{Mg}|} = \frac{\int h(y)dy}{M \int g(y)dy} = \frac{\int h(y)dy}{M}$$
  since g is a probability density. p is the *acceptance probability*.
- the expected number of draws needed is
  $$E[\text{number of draws}] = \frac{1}{p} = \frac{M \int g(y)dy}{\int h(y)dy} = \frac{M}{\int h(y)dy}$$
- if h is also a proper density, then $p = 1/M$ and
  $$E[\text{number of draws}] = \frac{1}{p} = M$$

The Basic Algorithm

The rejection, or accept/reject, sampling algorithm:

repeat
  generate independent $X \sim g$ and $U \sim \mathrm{U}[0,1]$
until $UMg(X) \leq h(X)$
return X

Alternate forms of the test:

$$U \leq \frac{h(X)}{Mg(X)}$$
$$\log(U) \leq \log(h(X)) - \log(M) - \log(g(X))$$

Care may be needed to ensure numerical stability.

Example: Normal Distribution with Cauchy Envelope

Suppose

- $h(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}$ is the standard normal density
- $g(x) = \frac{1}{\pi(1+x^2)}$ is the standard Cauchy density

Then

$$\frac{h(x)}{g(x)} = \sqrt{\frac{\pi}{2}}(1+x^2)e^{-x^2/2} \leq \sqrt{\frac{\pi}{2}}(1+1^2)e^{-1^2/2} = \sqrt{2\pi e^{-1}} = 1.520347$$

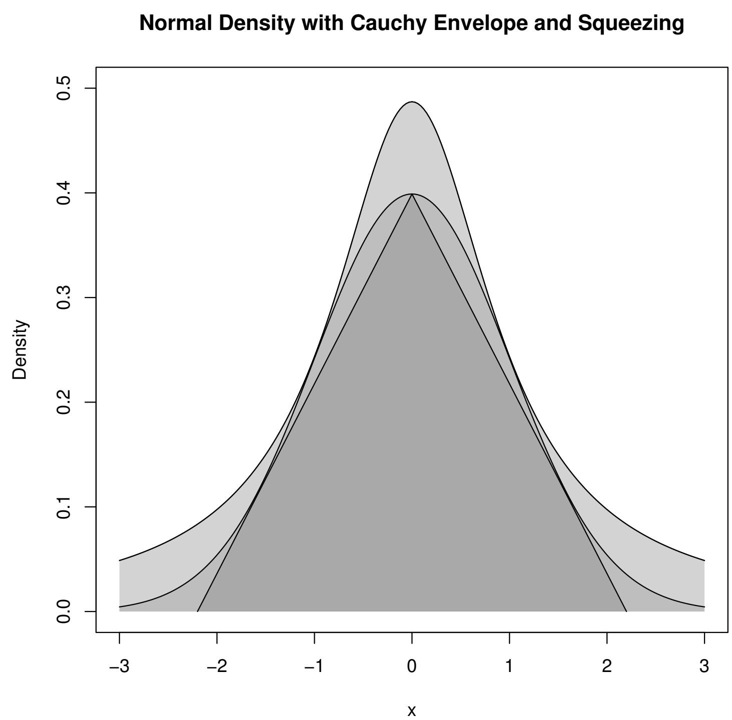
The resulting accept/reject algorithm is

repeat
  generate independent standard Cauchy X and $U \sim \mathrm{U}[0,1]$
until $U \leq \frac{e^{1/2}}{2}(1+X^2)e^{-X^2/2}$
return X

Squeezing

Performance can be improved by *squeezing*:

- Accept if point is inside the triangle:



- Squeezing *can* speed up generation.
- Squeezing *will* complicate the code (making errors more likely).

Rejection Sampling for Discrete Distributions

For simplicity, just consider integer valued random variables.

- If h and g are probability mass functions on the integers and $h(x)/g(x)$ is bounded, then the same algorithm can be used.
- If p is a probability mass function on the integers then

  $$h(x) = p(\lfloor x \rfloor)$$

  is a probability density.

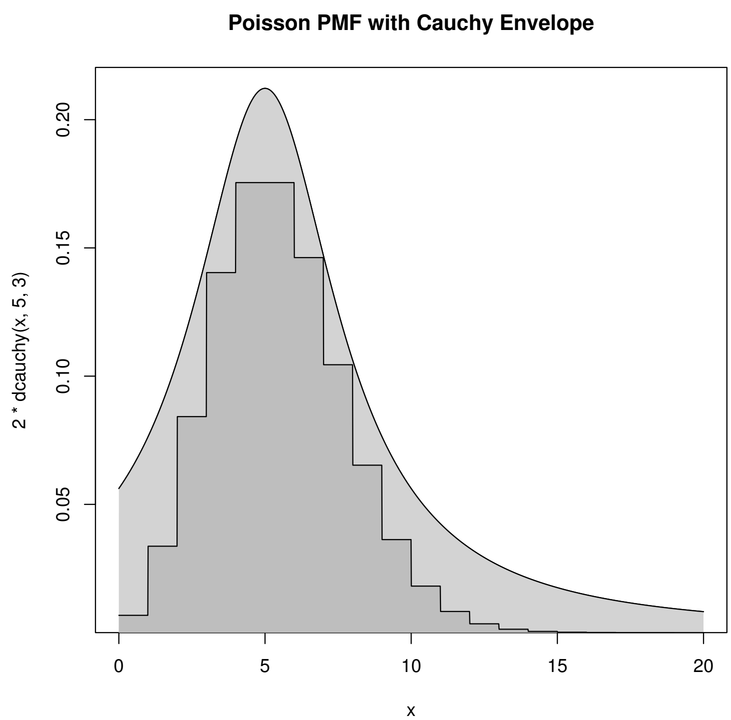
  If X has density h, then $Y = \lfloor X \rfloor$ has PMF p.

Example: Poisson Distribution with Cauchy Envelope

Suppose

- p is the PMF of a Poisson distribution with mean 5
- g is the Cauchy density with location 5 and scale 3.
- $h(x) = p(\lfloor x \rfloor)$

Then, by careful analysis or graphical examination, $h(x) \leq 2g(x)$ for all x.



Comments

- The Cauchy density is often a useful envelope.
- More efficient choices are often possible.
- Location and scale need to be chosen appropriately.
- If the target distribution is non-negative, a truncated Cauchy can be used.
- Careful analysis is needed to produce generators for a parametric family (e.g. all Poisson distributions).
- Graphical examination can be very helpful in guiding the analysis.
- Carefully tuned envelopes combined with squeezing can produce very efficient samplers.
- Errors in tuning and squeezing will produce garbage.

Ratio-of-Uniforms Method

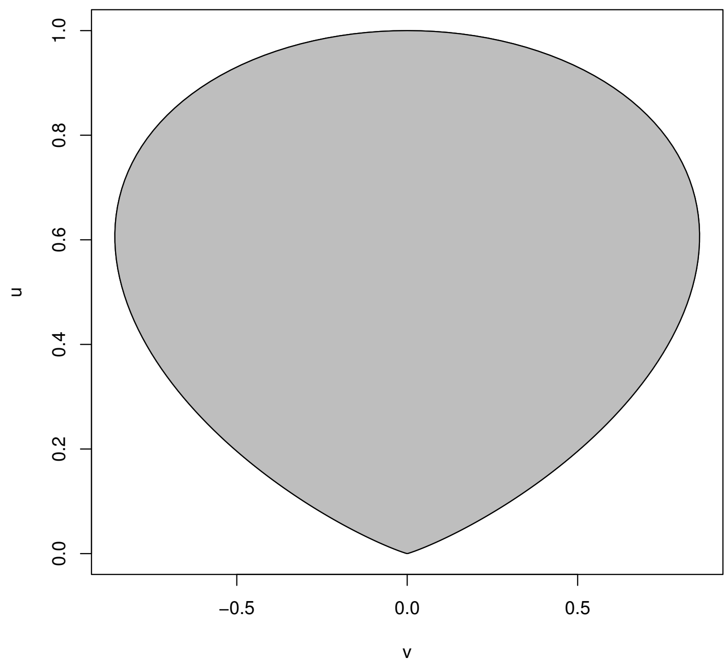
Basic Method

- Introduced by Kinderman and Monahan (1977).
- Suppose
  - $h(x) \geq 0$ for all x
  - $\int h(x)dx < \infty$
- Let (V,U) be uniform on

$$\mathscr{C}_h = \{(v,u) : 0 < u \leq \sqrt{h(v/u)}\}$$

  Then $X = V/U$ has density $f(x) = h(x)/\int h(y)dy$.

- For $h(x) = e^{-x^2/2}$ the region $\mathscr{C}_h$ looks like



  - The region is bounded.
  - The region is convex.

Properties

- The region $\mathscr{C}_h$ is convex if h is log concave.
- The region $\mathscr{C}_h$ is bounded if $h(x)$ and $x^2 h(x)$ are bounded.
- Let

$$\begin{aligned}
u^* &= \max_x \sqrt{h(x)}\\
v_-^* &= \min_x x\sqrt{h(x)}\\
v_+^* &= \max_x x\sqrt{h(x)}
\end{aligned}$$

  Then $\mathscr{C}_h$ is contained in the rectangle $[v_-^*, v_+^*] \times [0, u^*]$.

- The simple Ratio-of-Uniforms algorithm based on rejection sampling from the enclosing rectangle is

  repeat
  generate $U \sim \mathrm{U}[0, u^*]$
  generate $V \sim \mathrm{U}[v_-^*, v_+^*]$
  until $U^2 \leq h(V/U)$
  return $X = V/U$

- If $h = e^{-x^2/2}$ then

$$\begin{aligned}
u^* &= 1\\
v_-^* &= -\sqrt{2e^{-1}}\\
v_+^* &= \sqrt{2e^{-1}}
\end{aligned}$$

  and the expected number of draws is

$$\frac{\text{area of rectangle}}{\text{area of } \mathscr{C}_h} = \frac{u^*(v_+^* - v_-^*)}{\frac{1}{2}\int h(x)dx} = \frac{2\sqrt{2e^{-1}}}{\sqrt{\pi/2}} = 1.368793$$

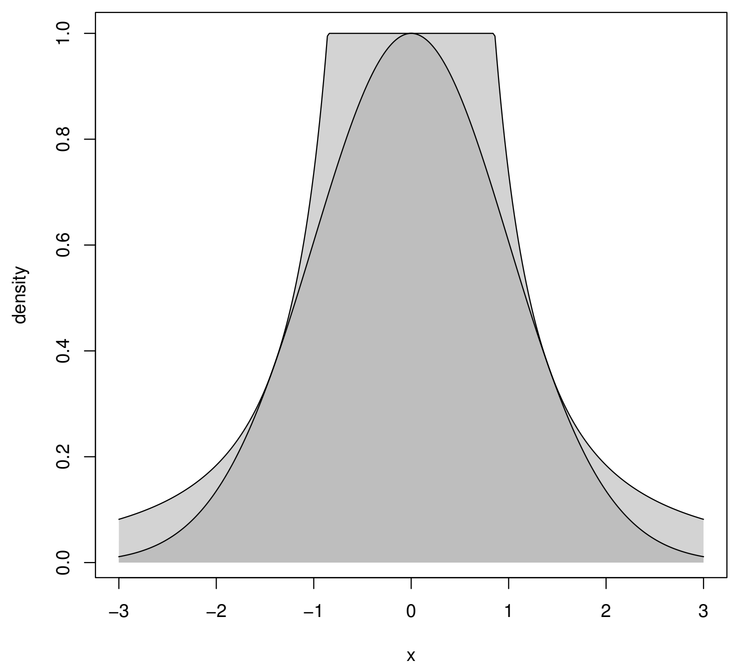
- Various squeezing methods are possible.
- Other approaches to sampling from $\mathscr{C}_h$ are also possible.

Relation to Rejection Sampling

Ratio of Uniforms with rejection sampling from the enclosing rectangle is equivalent to ordinary rejection sampling using an envelope density

$$g(x) \propto \begin{cases} \left(\frac{v_-^*}{x}\right)^2 & \text{if } x < v_-^*/u^* \\ (u^*)^2 & \text{if } v_-^*/u^* \le x \le v_+^*/u^* \\ \left(\frac{v_+^*}{x}\right)^2 & \text{if } x > v_+^*/u^* \end{cases}$$

This is sometimes called a *table mountain density*



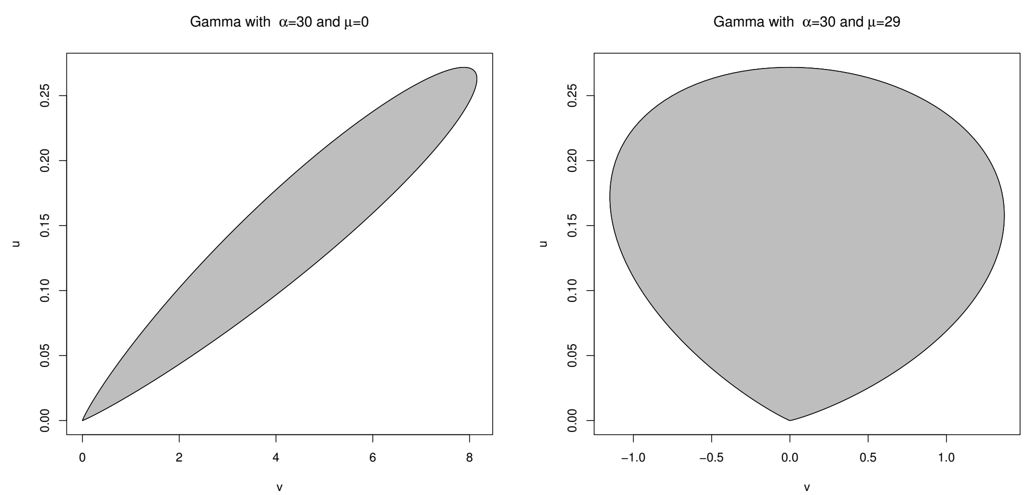
Generalizations

A more general form of the basic result: For any μ and any $r > 0$ let

$$\mathscr{C}_{h,\mu,r} = \{(v,u) : 0 < u \leq h(v/u^r + \mu)^{1/(r+1)}\}$$

If (U,V) is uniform on $\mathscr{C}_{h,\mu,r}$, then $X = V/U^r + \mu$ has density $f(x) = h(x)/\int h(y)dy$.

- μ and r can be chosen to minimize the rejection probability.
- $r = 1$ seems adequate for most purposes.
- Choosing μ equal to the mode of h can help.
- For the Gamma distribution with $\alpha = 30$,



Adaptive Rejection Sampling

First introduced by Gilks and Wild (1992).

Convexity

- A set C is convex if $\lambda x + (1-\lambda)y \in C$ for all $x, y \in C$ and $\lambda \in [0,1]$.
- C can be a subset or $\mathbb{R}$, or $\mathbb{R}^n$, or any other set where the *convex combination*

  $$\lambda x + (1-\lambda)y$$

  makes sense.
- A real-valued function f on a convex set C is convex if

  $$f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda) f(y)$$

  $x, y \in C$ and $\lambda \in [0,1]$.
- $f(x)$ is concave if $-f(x)$ is convex, i.e. if

  $$f(\lambda x + (1-\lambda)y) \geq \lambda f(x) + (1-\lambda) f(y)$$

  $x, y \in C$ and $\lambda \in [0,1]$.
- A concave function is always below its tangent.

Log Concave Densities

- A density f is *log concave* if $\log f$ is a concave function
- Many densities are log concave:
  - normal densities
  - Gamma(α, β) with $\alpha \geq 1$
  - Beta(α, β) with $\alpha \geq 1$ and $\beta \geq 1$.
- Some are not but may be related to ones that are: The t densities are not, but if

$$\begin{aligned} X|Y = y &\sim \mathrm{N}(0, 1/y) \\ Y &\sim \mathrm{Gamma}(\alpha, \beta) \end{aligned}$$

  then

  - the marginal distribution of X is t for suitable choice of β
  - and the joint distribution of X and Y has density

$$f(x,y) \propto \sqrt{y} e^{-\frac{y}{2}x^2} y^{\alpha-1} e^{-y/\beta} = y^{\alpha-1/2} e^{-y(\beta+x^2/2)}$$

    which is log concave for $\alpha \geq 1/2$

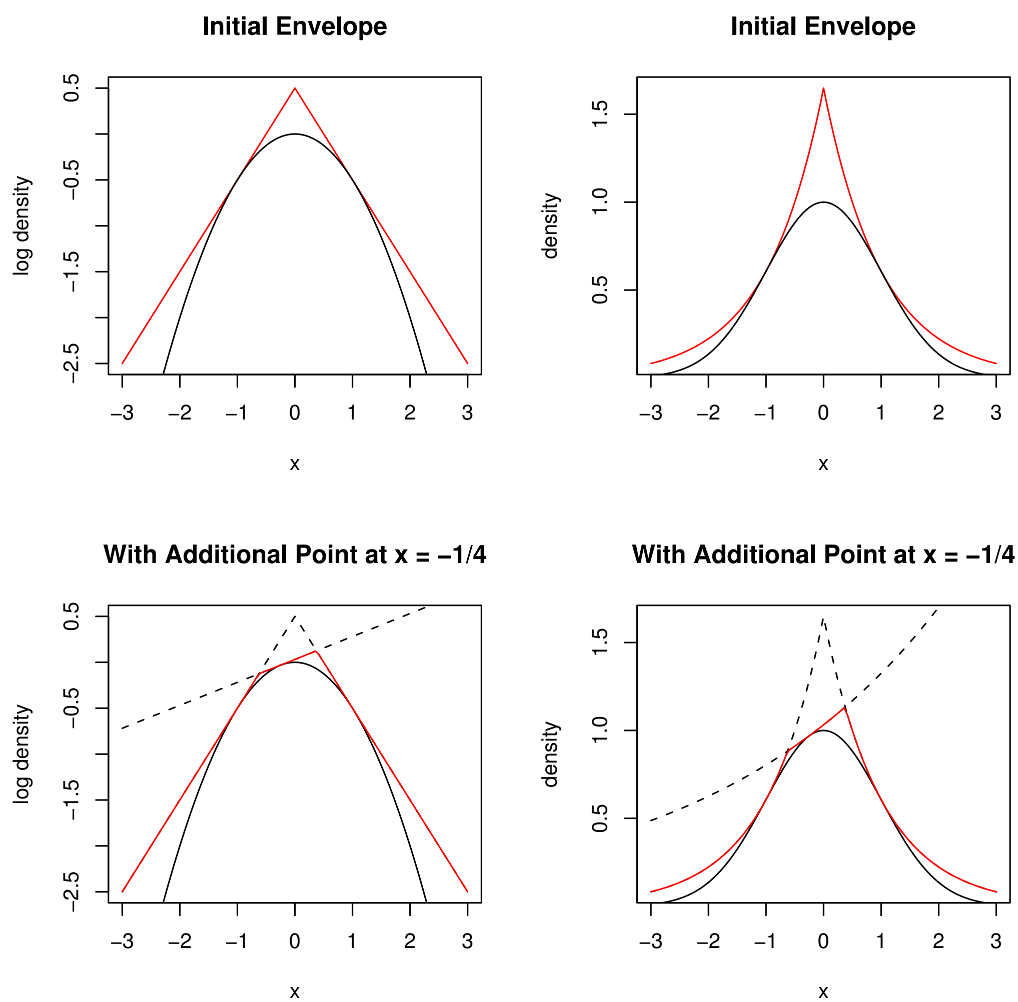
Tangent Approach

Suppose

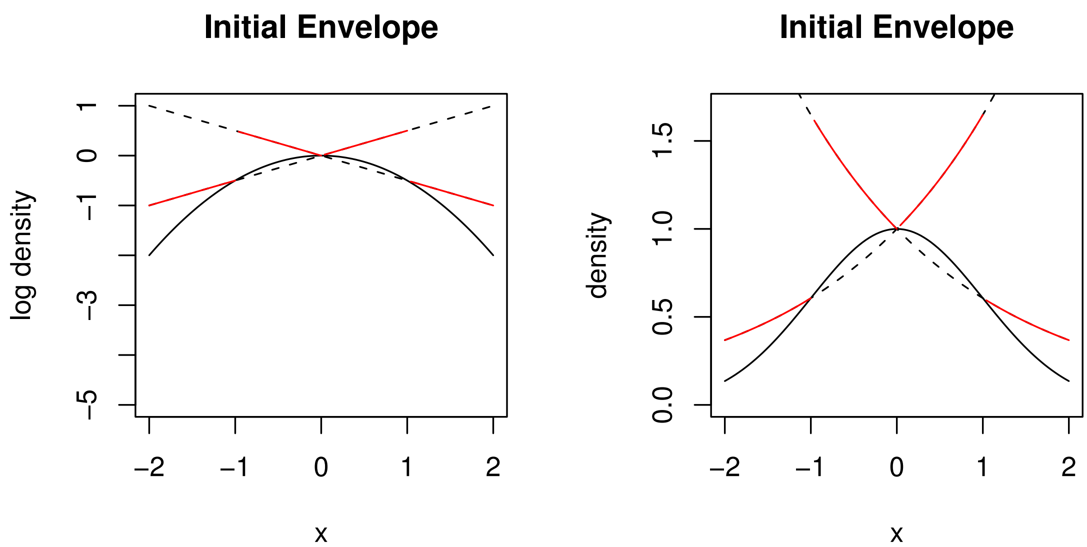
- f is log concave
- f has an interior mode

Need log density, derivative at two points, one each side of the mode

- piece-wise linear envelope of log density
- piece-wise exponential envelope of density
- if first point is not accepted, can use to make better envelope



Secant Approach



- Need three points to start
- Do not need derivatives
- Get larger rejection rates
- Both approaches need numerical care

Notes and Comments

- Many methods depend on properties of a particular distribution.
- Inversion is one general method that can often be used.
- Other general-purpose methods are
  - rejection sampling
  - adaptive rejection sampling
  - ratio-of-uniforms
- Some references:
  - Devroye, L. (1986). *Non-Uniform Random Variate Generation*, Springer-Verlag, New York.
  - Gentle, J. E. (2003). *Random Number Generation and Monte Carlo Methods*, Springer-Verlag, New York.
  - Hörmann, W., Leydold, J., and Derflinger, G. (2004). *Automatic Nonuniform Random Variate Generation*, Springer-Verlag, New York.

A Recent Publication

Karney, C.F.F. (2016). "Sampling Exactly from the Normal Distribution." *ACM Transactions on Mathematical Software* 42 (1).

Random Variate Generators in R

- Generators for most standard distributions are available
  - `rnorm`: normal
  - `rgamma`: gamma
  - `rt`: t
  - `rpois`: Poisson
  - etc.
- Most use standard algorithms from the literature.
- Source code is in `src/nmath/` in the source tree,

  `https://svn.r-project.org/R/trunk`

- The normal generator can be configured by `RNGkind`. Options are
  - Kinderman-Ramage
  - Buggy Kinderman-Ramage (available for reproducing results)
  - Ahrens-Dieter
  - Box-Muller
  - Inversion (the current default)
  - user-supplied

Generating Random Vectors and Matrices

- Sometimes generating random vectors can be reduced to a series of univariate generations.
- One approach is conditioning:

$$f(x,y,z) = f_{Z|X,Y}(z|x,y)f_{Y|X}(y|x)f_X(x)$$

  So we can generate

  - X from $f_X(x)$
  - $Y|X=x$ from $f_{Y|X}(y|x)$
  - $Z|X=x,Y=y$ from $f_{Z|X,Y}(z|x,y)$

- One example: $(X_1,X_2,X_3) \sim \text{Multinomial}(n,p_1,p_2,p_3)$ Then

$$\begin{aligned}
X_1 &\sim \text{Binomial}(n,p_1)\\
X_2|X_1=x_1 &\sim \text{Binomial}(n-x_1,p_2/(p_2+p_3))\\
X_3|X_1=x_1,X_2=x_2 &= n-x_1-x_2
\end{aligned}$$

- Another example: X,Y bivariate normal $(\mu_X,\mu_Y,\sigma_X^2,\sigma_Y^2,\rho)$. Then

$$\begin{aligned}
X &\sim \text{N}(\mu_X,\sigma_X^2)\\
Y|X=x &\sim \text{N}\left(\mu_Y+\rho\frac{\sigma_Y}{\sigma_X}(x-\mu_X),\sigma_Y^2(1-\rho^2)\right)
\end{aligned}$$

- For some distributions special methods are available.
- Some general methods extend to multiple dimensions.

Multivariate Normal Distribution

- Marginal and conditional distributions are normal; conditioning can be used in general.
- Alternative: use linear transformations.

  Suppose $Z_1, \ldots, Z_d$ are independent standard normals, $\mu_1, \ldots \mu_d$ are constants, and A is a constant $d \times d$ matrix. Let

  $$Z = \begin{bmatrix} Z_1 \\ \vdots \\ Z_d \end{bmatrix} \qquad \mu = \begin{bmatrix} \mu_1 \\ \vdots \\ \mu_d \end{bmatrix}$$

  and set

  $$X = \mu + AZ$$

  Then X is multivariate normal with mean vector μ and covariance matrix AA^T,

  $$X \sim \mathrm{MVN}_d(\mu, AA^T)$$

- To generate $X \sim \mathrm{MVN}_d(\mu, \Sigma)$, we can
  - find a matrix A such that $AA^T = \Sigma$
  - generate elements of Z as independent standard normals
  - compute $X = \mu + AZ$
- The Cholesky factorization is one way to choose A.
- If we are given Σ^{-1}, then we can
  - decompose $\Sigma^{-1} = LL^T$
  - solve $L^T Y = Z$
  - compute $X = \mu + Y$

Spherically Symmetric Distributions

- A joint distribution with density of the form
$$f(x) = g(x^T x) = g(x_1^2 + \cdots + x_d^2)$$
is called spherically symmetric (about the origin).

- If the distribution of X is spherically symmetric then
$$\begin{aligned} R &= \sqrt{X^T X} \\ Y &= X/R \end{aligned}$$
are independent,
  - Y is uniformly distributed on the surface of the unit sphere.
  - R has density proportional to $g(r)r^{d-1}$ for $r > 0$.

- We can generate $X \sim f$ by
  - generating $Z \sim \text{MVN}_d(0, I)$ and setting $Y = Z/\sqrt{Z^T Z}$
  - generating R from the density proportional to $g(r)r^{d-1}$ by univariate methods.

Elliptically Contoured Distributions

- A density f is elliptically contoured if
$$f(x) = \frac{1}{\sqrt{\det \Sigma}} g((x-\mu)^T \Sigma^{-1}(x-\mu))$$
for some vector μ and symmetric positive definite matrix Σ.

- Suppose Y has spherically symmetric density $g(y^T y)$ and $AA^T = \Sigma$. Then $X = \mu + AY$ has density f.

Wishart Distribution

- Suppose $X_1, \ldots X_n$ are independent and $X_i \sim \mathrm{MVN}_d(\mu_i, \Sigma)$. Let

$$W = \sum_{i=1}^{n} X_i X_i^T$$

  Then W has a non-central Wishart distribution $\mathrm{W}(n, \Sigma, \Delta)$ where $\Delta = \sum \mu_i \mu_i^T$.

- If $X_i \sim \mathrm{MVN}_d(\mu, \Sigma)$ and

$$S = \frac{1}{n-1} \sum_{i=1}^{n} (X_i - \overline{X})(X_i - \overline{X})^T$$

  is the sample covariance matrix, then $(n-1)S \sim \mathrm{W}(n-1, \Sigma, 0)$.

- Suppose $\mu_i = 0$, $\Sigma = AA^T$, and $X_i = AZ_i$ with $Z_i \sim \mathrm{MVN}_d(0, I)$. Then $W = AVA^T$ with

$$V = \sum_{i=1}^{n} Z_i Z_i^T$$

- *Bartlett decomposition*: In the Cholesky factorization of V
  - all elements are independent
  - the elements below the diagonal are standard normal
  - the square of the i-th diagonal element is χ^2_{n+1-i}

- If $\Delta \neq 0$ let $\Delta = BB^T$ be its Cholesky factorization, let b_i be the columns of B and let $Z_1, \ldots, Z_n$ be independent $\mathrm{MVN}_d(0, I)$ random vectors. Then for $n \geq d$

$$W = \sum_{i=1}^{d} (b_i + AZ_i)(b_i + AZ_i)^T + \sum_{i=d+1}^{n} AZ_i Z_i^T A^T \sim \mathrm{W}(n, \Sigma, \Delta)$$

Rejection Sampling

- Rejection sampling can in principle be used in any dimensions
- A general envelope that is sometimes useful is based on generating X as

  $$X = b + AZ/Y$$

  where

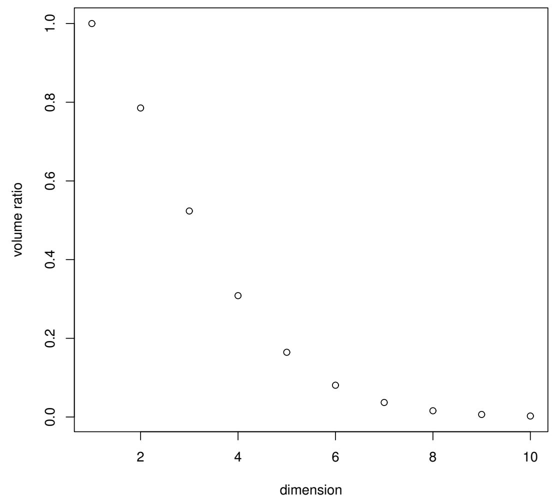
  - Z and Y are independent
  - $Z \sim \mathrm{MVN}_d(0, I)$
  - $Y^2 \sim \mathrm{Gamma}(\alpha, 1/\alpha)$, a scalar
  - b is a vector of constants
  - A is a matrix of constants

  This is a kind of multivariate t random vector.
- This often works in modest dimensions.
- Specially tailored envelopes can sometimes be used in higher dimensions.
- Without special tailoring, rejection rates tend to be too high to be useful.

Ratio of Uniforms

- The ratio-of-uniforms method also works in $\mathbb{R}^d$: Suppose
  - $h(x) \geq 0$ for all x
  - $\int h(x)dx < \infty$

  Let
  $$\mathscr{C}_h = \{(v,u) : v \in \mathbb{R}^d, 0 < u \leq \sqrt[d+1]{h(v/u+\mu)}\}$$
  for some μ. If (V,U) is uniform on $\mathscr{C}_h$, then $X = V/U + \mu$ has density $f(x) = h(x)/\int h(y)dy$.
- If $h(x)$ and $\|x\|^{d+1}h(x)$ are bounded, then $\mathscr{C}_h$ is bounded.
- If $h(x)$ is log concave then $\mathscr{C}_h$ is convex.
- Rejection sampling from a bounding hyper rectangle works in modest dimensions.
- It will not work for dimensions larger than 8 or so:
  - The shape of $\mathscr{C}_h$ is vaguely spherical.
  - The volume of the unit sphere in d dimensions is
    $$V_d = \frac{\pi^{d/2}}{\Gamma(d/2+1)}$$
  - The ratio of this volume to the volume of the enclosing hyper cube, 2^d tends to zero very fast:


volume ratio
0.0
0.2
0.4
0.6
0.8
1.0
2
4
6
8
10
dimension

Order Statistics

- The order statistics for a random sample $X_1, \dots, X_n$ from F are the ordered values
$$X_{(1)} \le X_{(2)} \le \cdots \le X_{(n)}$$
  - We can simulate them by ordering the sample.
  - Faster $O(n)$ algorithms are available for individual order statistics, such as the median.
- If $U_{(1)} \le \cdots \le U_{(n)}$ are the order statistics of a random sample from the $\mathrm{U}[0,1]$ distribution, then
$$\begin{aligned} X_{(1)} &= F^{-}(U_{(1)}) \\ &\vdots \\ X_{(n)} &= F^{-}(U_{(n)}) \end{aligned}$$
are the order statistics of a random sample from F.
- For a sample of size n the marginal distribution of $U_{(k)}$ is
$$U_{(k)} \sim \mathrm{Beta}(k, n-k+1).$$
- Suppose $k < \ell$.
  - Then $U_{(k)}/U_{(\ell)}$ is independent of $U_{(\ell)}, \dots, U_{(n)}$
  - $U_{(k)}/U_{(\ell)}$ has a $\mathrm{Beta}(k, \ell - k)$ distribution.

  We can use this to generate any subset or all order statistics.
- Let $V_1, \dots, V_{n+1}$ be independent exponential random variables with the same mean and let
$$W_k = \frac{V_1 + \cdots + V_k}{V_1 + \cdots + V_{n+1}}$$
Then $W_1, \dots, W_n$ has the same joint distribution as $U_{(1)}, \dots, U_{(n)}$.

Homogeneous Poisson Process

- For a homogeneous Poisson process with rate λ
  - The number of points $N(A)$ in a set A is Poisson with mean $\lambda|A|$.
  - If A and B are disjoint then $N(A)$ and $N(B)$ are independent.
- Conditional on $N(A) = n$, the n points are uniformly distributed on A.
- We can generate a Poisson process on $[0,t]$ by generating exponential variables $T_1, T_2, \ldots$ with rate λ and computing

  $$S_k = T_1 + \cdots + T_k$$

  until $S_k > t$. The values $S_1, \ldots, S_{k-1}$ are the points in the Poisson process realization.

Inhomogeneous Poisson Processes

- For an inhomogeneous Poisson process with rate $\lambda(x)$
  - The number of points $N(A)$ in a set A is Poisson with mean $\int_A \lambda(x)dx$.
  - If A and B are disjoint then $N(A)$ and $N(B)$ are independent.
- Conditional on $N(A) = n$, the n points in A are a random sample from a distribution with density $\lambda(x)/\int_A \lambda(y)dy$.
- To generate an inhomogeneous Poisson process on $[0,t]$ we can
  - let $\Lambda(s) = \int_0^s \lambda(x)dx$
  - generate arrival times $S_1,\ldots,S_N$ for a homogeneous Poisson process with rate one on $[0,\Lambda(t)]$
  - Compute arrival times of the inhomogeneous process as
  $$\Lambda^{-1}(S_1),\ldots,\Lambda^{-1}(S_N).$$
- If $\lambda(x) \leq M$ for all x, then we can generate an inhomogeneous Poisson process with rate $\lambda(x)$ by *thinning*:
  - generate a homogeneous Poisson process with rate M to obtain points $X_1,\ldots,X_N$.
  - independently delete each point X_i with probability $1-\lambda(X_i)/M$.

  The remaining points form a realization of an inhomogeneous Poisson process with rate $\lambda(x)$.
- If N_1 and N_2 are independent inhomogeneous Poisson processes with rates $\lambda_1(x)$ and $\lambda_2(x)$, then their superposition $N_1 + N_2$ is an inhomogeneous Poisson process with rate $\lambda_1(x)+\lambda_2(x)$.

Other Processes

- Many other processes can be simulated from their definitions
  - Cox processes (doubly stochastic Poisson process)
  - Poisson cluster processes
  - ARMA, ARIMA processes
  - GARCH processes
- Continuous time processes, such as Brownian motion and diffusions, require discretization of time.
- Other processes may require Markov chain methods
  - Ising models
  - Strauss process
  - interacting particle systems

Variance Reduction

Most simulations involve estimating integrals or expectations:

$$
\begin{aligned}
\theta &= \int h(x) f(x) dx && \text{mean} \\
\theta &= \int 1_{\{X \in A\}} f(x) dx && \text{probability} \\
\theta &= \int (h(x) - E[h(X)])^2 f(x) dx && \text{variance} \\
&\vdots
\end{aligned}
$$

- The *crude simulation*, or *crude Monte Carlo*, or *naïve Monte Carlo*, approach:
  - Sample $X_1, \dots, X_N$ independently from f
  - Estimate θ by $\widehat{\theta}_N = \frac{1}{N} \sum h(X_i)$.

  If $\sigma^2 = \text{Var}(h(X))$, then $\text{Var}(\widehat{\theta}_N) = \frac{\sigma^2}{N}$.
- To reduce the error we can
  - increase N: requires CPU time and clock time; diminishing returns.
  - try to reduce σ^2: requires thinking time, programming effort.
- Methods that reduce σ^2 are called
  - tricks
  - swindles
  - Monte Carlo methods

Control Variates

Suppose we have a random variable Y with mean θ and a correlated random variable W with known mean $E[W]$. Then for any constant b

$$\widetilde{Y} = Y - b(W - E[W])$$

has mean θ.

- W is called a *control variate*.
- Choosing $b = 1$ often works well if the correlation is positive and θ and $E[W]$ are close.
- The value of b that minimizes the variance of $\widetilde{Y}$ is $\text{Cov}(Y,W)/\text{Var}(W)$.
- We can use a guess or a pilot study to estimate b.
- We can also estimate b from the same data used to compute Y and W.
- This is related to the regression estimator in sampling.

Example

Suppose we want to estimate the expected value of the sample median T for a sample of size 10 from a Gamma$(3,1)$ population.

- Crude estimate:
$$Y = \frac{1}{N}\sum T_i$$
- Using the sample mean as a control variate with $b = 1$:
$$\widehat{Y} = \frac{1}{N}\sum(T_i - \overline{X}_i) + E[\overline{X}_i] = \frac{1}{N}\sum(T_i - \overline{X}_i) + \alpha$$

```
> x <- matrix(rgamma(10000, 3), ncol = 10)
> md <- apply(x, 1, median)
> mn <- apply(x, 1, mean)
> mean(md)
[1] 2.711137
> mean(md - mn) + 3
[1] 2.694401
> sd(md)
[1] 0.6284996
> sd(md-mn)
[1] 0.3562479
```

The standard deviation is cut roughly in half. The optimal b seems close to 1.

Control Variates and Probability Estimates

- Suppose T is a test statistic and we want to estimate $\theta = P(T \leq t)$.
- Crude Monte Carlo:
$$\widehat{\theta} = \frac{\#\{T_i \leq t\}}{N}$$
- Suppose S is "similar" to T and $P(S \leq t)$ is known. Use
$$\widehat{\theta} = \frac{\#\{T_i \leq t\} - \#\{S_i \leq t\}}{N} + P(S \leq t) = \frac{1}{N}\sum Y_i + P(S \leq t)$$
with $Y_i = 1_{\{T_i \leq t\}} - 1_{\{S_i \leq t\}}$.
- If S mimics T, then Y_i is usually zero.
- Could use this to calibrate
$$T = \frac{\text{median}}{\text{interquartile range}}$$
for normal data using the t statistic.

Importance Sampling

- Suppose we want to estimate

  $$\theta = \int h(x)f(x)dx$$

  for some density f and some function h.

- Crude Mote Carlo samples $X_1,\ldots,X_N$ from f and uses

  $$\widehat{\theta} = \frac{1}{N}\sum h(X_i)$$

  If the region where h is large has small probability under f then this can be inefficient.

- Alternative: Sample $X_1,\ldots X_n$ from g that puts more probability near the "important" values of x and compute

  $$\widetilde{\theta} = \frac{1}{N}\sum h(X_i)\frac{f(X_i)}{g(X_i)}$$

  Then, if $g(x) > 0$ when $h(x)f(x) \neq 0$,

  $$E[\widetilde{\theta}] = \int h(x)\frac{f(x)}{g(x)}g(x)dx = \int h(x)f(x)dx = \theta$$

  and

  $$\mathrm{Var}(\widetilde{\theta}) = \frac{1}{N}\int\left(h(x)\frac{f(x)}{g(x)} - \theta\right)^2 g(x)dx = \frac{1}{N}\left(\int\left(h(x)\frac{f(x)}{g(x)}\right)^2 g(x)dx - \theta^2\right)$$

  The variance is minimized by $g(x) \propto |h(x)f(x)|$

Importance Weights

- Importance sampling is related to stratified and weighted sampling in sampling theory.
- The function $w(x) = f(x)/g(x)$ is called the *weight function*.
- Alternative estimator:

$$\theta^* = \frac{\sum h(X_i)w(X_i)}{\sum w(X_i)}$$

  This is useful if f or g or both are unnormalized densities.
- Importance sampling can be useful for computing expectations with respect to posterior distributions in Bayesian analyses.
- Importance sampling can work very well if the weight function is bounded.
- Importance sampling can work very poorly if the weight function is unbounded—it is easy to end up with infinite variances.

Computing Tail Probabilities

- Suppose $\theta = P(X \in R)$ for some region R.
- Suppose we can find g such that $f(x)/g(x) < 1$ on R. Then

$$\widetilde{\theta} = \frac{1}{N}\sum 1_R(X_i)\frac{f(X_i)}{g(X_i)}$$

  and

$$\begin{aligned}
\mathrm{Var}(\widetilde{\theta}) &= \frac{1}{N}\left(\int_R \left(\frac{f(x)}{g(x)}\right)^2 g(x)dx - \theta^2\right)\\
&= \frac{1}{N}\left(\int_R \frac{f(x)}{g(x)} f(x)dx - \theta^2\right)\\
&< \frac{1}{N}\left(\int_R f(x)dx - \theta^2\right)\\
&= \frac{1}{N}(\theta - \theta^2) = \mathrm{Var}(\widehat{\theta})
\end{aligned}$$

- For computing $P(X > 2)$ where X has a standard Cauchy distribution we can use a shifted distribution:

```
> y <- rcauchy(10000,3)
> tt <- ifelse(y > 2, 1, 0) * dcauchy(y) / dcauchy(y,3)
> mean(tt)
[1] 0.1490745
> sd(tt)
[1] 0.1622395
```

- The asymptotic standard deviation for crude Monte Carlo is approximately

```
> sqrt(mean(tt) * (1 - mean(tt)))
[1] 0.3561619
```

- A *tilted* density $g(x) \propto f(x)e^{\beta x}$ can also be useful.

Antithetic Variates

- Suppose S and T are two unbiased estimators of θ with the same variance σ^2 and correlation ρ, and compute

$$V = \frac{1}{2}(S+T)$$

  Then

$$\text{Var}(V) = \frac{\sigma^2}{4}(2+2\rho) = \frac{\sigma^2}{2}(1+\rho)$$

- Choosing $\rho < 0$ reduces variance.
- Such negatively correlated pairs are called *antithetic variates*.
- Suppose we can choose between generating independent $T_1,\ldots,T_N$

$$\widehat{\theta} = \frac{1}{N}\sum_{i=1}^{N} T_i$$

  or independent pairs $(S_1,T_1),\ldots,(S_{N/2},T_{N/2})$ and computing

$$\widetilde{\theta} = \frac{1}{N}\sum_{i=1}^{N/2}(S_i+T_i)$$

  If $\rho < 0$, then $\text{Var}(\widetilde{\theta}) < \text{Var}(\widehat{\theta})$.

- If $T = f(U)$, $U \sim \text{U}[0,1]$, and f is monotone, then $S = f(1-U)$ is negatively correlated with T and has the same marginal distribution.
- If inversion is used to generate variates, computing T from $U_1,\ldots$ and S from $1-U_1,\ldots$ often works.
- Some uniform generators provide an option in the seed to switch between returning U_i and $1-U_i$.

Example

For estimating the expected value of the median for samples of size 10 from the Gamma$(3,1)$ distribution:

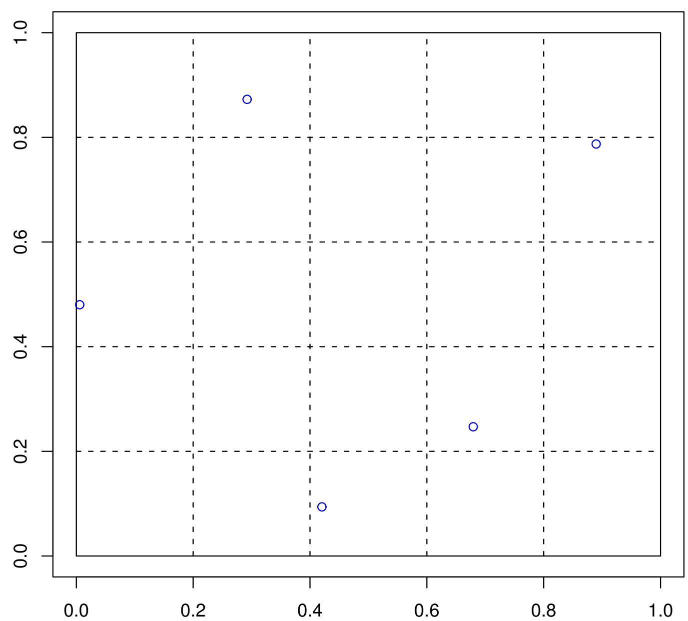
```
> u <- matrix(runif(5000), ncol = 10)
> x1 <- qgamma(u, 3)
> x2 <- qgamma(1 - u, 3)
> md1 <- apply(x1, 1, median)
> md2 <- apply(x2, 1, median)
> sqrt(2) * sd((md1 + md2) / 2)
[1] 0.09809588
```

Control variates helps further a bit but need $b = 0.2$ or so.

```
> mn1 <- apply(x1, 1, mean)
> mn2 <- apply(x2, 1, mean)
> sqrt(2) * sd((md1 + md2 - 0.2 * (mn1 + mn2)) / 2)
[1] 0.09216334
```

Latin Hypercube Sampling

- Suppose we want to compute
$$\theta = E[f(U_1,\dots,U_d)]$$
with $(U_1,\dots,U_d)$ uniform on $[0,1]^d$.
- For each i
  - independently choose permutation π_i of $\{1,\dots,N\}$
  - generate $U_i^{(j)}$ uniformly on $[\pi_i(j)/N, (\pi_i(j)+1)/N]$.
- For $d = 2$ and $N = 5$:



This is a random Latin square design.

- In many cases this reduces variance compared to unrestricted random sampling (Stein, 1987; Avramidis and Wilson, 1995; Owen, 1992, 1998)

Common Variates and Blocking

- Suppose we want to estimate $\theta = E[S] - E[T]$
- One approach is to chose independent samples $T_1, \ldots, T_N$ and $S_1, \ldots, S_M$ and compute
$$\widehat{\theta} = \frac{1}{M}\sum_{i=1}^{M} S_i - \frac{1}{N}\sum_{i=1}^{N} T_i$$
- Suppose $S = S(X)$ and $T = T(X)$ for some X. Instead of generating independent X values for S and T we may be able to
  - use the common X values to generate pairs $(S_1, T_1), \ldots, (S_N, T_N)$
  - compute
$$\widetilde{\theta} = \frac{1}{N}\sum_{i=1}^{N} (S_i - T_i)$$
- This use of *paired comparisons* is a form of *blocking*.
- This idea extends to comparisons among more than two statistics.
- In simulations, we can often do this by using the same random variates to generate S_i and T_i. This is called using *common variates*.
- This is easiest to do if we are using inversion; this, and the ability to use antithetic variates, are two strong arguments in favor of inversion.
- Using common variates may be harder when rejection-based methods are involved.
- In importance sampling, using
$$\theta^* = \frac{\sum h(X_i)w(X_i)}{\sum w(X_i)}$$
can be viewed as a paired comparison; for some forms of h is can have lower variance than the estimator that does not normalize by the sum of the weights.

Conditioning or Rao-Blackwellization

- Suppose we want to estimate $\theta = E[X]$
- If X, W are jointly distributed, then

  $$\theta = E[X] = E[E[X|W]]$$

  and

  $$\mathrm{Var}(X) = E[\mathrm{Var}(X|W)] + \mathrm{Var}(E[X|W]) \geq \mathrm{Var}(E[X|W])$$

- Suppose we can compute $E[X|W]$. Then we can
  - generate $W_1, \ldots, W_N$
  - compute

    $$\widetilde{\theta} = \frac{1}{N} \sum E[X|W_i]$$

- This is often useful in Gibbs sampling.
- Variance reduction is not guaranteed if $W_1, \ldots, W_N$ are not independent.
- Conditioning is particularly useful for density estimation: If we can compute $f_{X|W}(x|w)$ and generate $W_1, \ldots, W_N$, then

  $$\widehat{f_X}(x) = \frac{1}{N} \sum f_{X|W}(x|W_i)$$

  is much more accurate than, say, a kernel density estimate based on a sample $X_1, \ldots, X_N$.

Example

Suppose we want to estimate $\theta = P(X > t)$ where $X = Z/W$ with Z, W independent, $Z \sim \mathrm{N}(0,1)$ and $W > 0$. Then

$$P(X > t | W = w) = P(Z > tw) = 1 - \Phi(tw)$$

So we can estimate θ by generating $W_1, \ldots, W_N$ and computing

$$\widetilde{\theta} = \frac{1}{N} \sum (1 - \Phi(tW_i))$$

Independence Decomposition

- Suppose $X_1,\dots,X_n$ is a random sample from a $\mathrm{N}(0,1)$ distribution and
$$\widetilde{X} = \text{median}(X_1,\dots,X_n)$$
We want to estimate $\theta = \text{Var}(\widetilde{X}) = E[\widetilde{X}^2]$.
- Crude Monte Carlo estimate: generate independent medians $\widetilde{X}_1,\dots,\widehat{X}_N$ and compute
$$\widehat{\theta} = \frac{1}{N}\sum \widetilde{X}_i^2$$
- Alternative: Write
$$\widetilde{X} = (\widetilde{X} - \overline{X}) + \overline{X}$$
$(\widetilde{X} - \overline{X})$ and $\overline{X}$ are independent, for example by Basu's theorem. So
$$E[\widetilde{X}^2|\overline{X}] = \overline{X}^2 + E[(\widetilde{X} - \overline{X})^2]$$
and
$$\theta = \frac{1}{n} + E[(\widetilde{X} - \overline{X})^2]$$
- So we can estimate θ by generating pairs $(\widetilde{X}_i, \overline{X}_i)$ and computing
$$\widetilde{\theta} = \frac{1}{n} + \frac{1}{N}\sum(\widetilde{X}_i - \overline{X}_i)^2$$
- Generating these pairs may be more costly than generating medians alone.

Example

```
> x <- matrix(rnorm(10000), ncol = 10)
> mn <- apply(x, 1, mean)
> md <- apply(x, 1, median)
> # estimates:
> mean(md^2)
[1] 0.1446236
> 1 / 10 + mean((md - mn)^2)
[1] 0.1363207
> # asymptotic standard errors:
> sd(md^2)
[1] 0.2097043
> sd((md - mn)^2)
[1] 0.0533576
```

Princeton Robustness Study

D. F. Andrews, P. J. Bickel, F. R. Hampel, P. J. Huber, W. H. Rogers, and J. W. Tukey, *Robustness of Location Estimates*, Princeton University Press, 1972.

- Suppose $X_1,\ldots,X_n$ are a random sample from a symmetric density

$$f(x-m).$$

- We want an estimator $T(X_1,\ldots,X_n)$ of m that is
  - accurate
  - robust (works well for a wide range of f's)
- Study considers many estimators, various different distributions.
- All estimators are unbiased and *affine equivariant*, i.e.

$$\begin{aligned} E[T] &= m \\ T(aX_1+b,\ldots,aX_n+b) &= aT(X_1,\ldots,X_n)+b \end{aligned}$$

  for any constants a,b. We can thus take $m=0$ without loss of generality.

Distributions Used in the Study

- Distributions considered were all of the form

$$X = Z/V$$

with $Z \sim \mathrm{N}(0,1)$, $V > 0$, and Z, V independent.

- Some examples:
  - $V \equiv 1$ gives $X \sim \mathrm{N}(0,1)$.
  - Contaminated normal:

$$V = \begin{cases} c & \text{with probability } \alpha \\ 1 & \text{with probability } 1-\alpha \end{cases}$$

  - Double exponential: $V \sim f_V(v) = v^{-3}e^{-v^{-2}/2}$
  - Cauchy: $V = |Y|$ with $Y \sim \mathrm{N}(0,1)$.
  - t_ν: $V \sim \sqrt{\chi^2_\nu/\nu}$.
- The conditional distribution $X|V = v$ is $\mathrm{N}(0, 1/v^2)$.
- Study generates X_i as Z_i/V_i.
- Write $X_i = \widehat{X} + \widehat{S}C_i$ with

$$\widehat{X} = \frac{\sum X_i V_i^2}{\sum V_i^2} \qquad \widehat{S}^2 = \frac{1}{n-1}\sum (X_i - \widehat{X})^2 V_i^2$$

Then

$$T(X) = \widehat{X} + \widehat{S}T(C)$$

- Can show that $\widehat{X}, \widehat{S}, C$ are conditionally independent given V.

Estimating Variances

- Suppose we want to estimate $\theta = \text{Var}(T) = E[T^2]$. Then
$$\begin{aligned}\theta &= E[(\widehat{X} + \widehat{S}T(C))^2]\\ &= E[\widehat{X}^2 + 2\widehat{S}\widehat{X}T(C) + \widehat{S}^2 T(C)^2]\\ &= E[E[\widehat{X}^2 + 2\widehat{S}\widehat{X}T(C) + \widehat{S}^2 T(C)^2 | V]]\end{aligned}$$
and
$$\begin{aligned}E[\widehat{X}^2|V] &= \frac{1}{\sum V_i^2}\\ E[\widehat{X}|V] &= 0\\ E[\widehat{S}^2|V] &= 1\end{aligned}$$
So
$$\theta = E\left[\frac{1}{\sum V_i^2}\right] + E[T(C)^2]$$
- Strategy:
  - Compute $E[T(C)^2]$ by crude Monte Carlo
  - Compute $E\left[\frac{1}{\sum V_i^2}\right]$ the same way or analytically.

  Exact calculations:
  - If $V_i \sim \sqrt{\chi^2_\nu/\nu}$, then
$$E\left[\frac{1}{\sum V_i^2}\right] = E\left[\frac{\nu}{\chi^2_{n\nu}}\right] = \frac{\nu}{n\nu - 2}$$
  - Contaminated normal:
$$E\left[\frac{1}{\sum V_i^2}\right] = \sum_{r=0}^{n} \binom{n}{r} \alpha^r (1-\alpha)^{n-r} \frac{1}{n - r + rc}$$

Comparing Variances

If T_1 and T_2 are two estimators, then
$$\text{Var}(T_1) - \text{Var}(T_2) = E[T_1(C)^2] - E[T_2(C)^2]$$
We can reduce variances further by using common variates.

Estimating Tail Probabilities

- Suppose we want to estimate

$$\begin{aligned}\theta &= P(T(X) > t)\\ &= P(\widehat{X} + \widehat{S}T(C) > t)\\ &= E\left[P\left(\sqrt{\sum V_i^2}\frac{t-\widehat{X}}{\widehat{S}} < \sqrt{\sum V_i^2}T(C)\middle| V,C\right)\right]\\ &= E\left[F_{t,n-1}\left(\sqrt{\sum V_i^2}T(C)\right)\right]\end{aligned}$$

  where $F_{t,n-1}$ is the CDF of a non-central t distribution (t is not the usual non-centrality parameter).

- This CDF can be evaluated numerically, so we can estimate θ by

$$\widehat{\theta}_N = \frac{1}{N}\sum_{k=1}^{N} F_{t,n-1}\left(T(C^{(k)})\sqrt{\sum {V_i^{(k)}}^2}\right)$$

- An alternative is to condition on $V, C, \widehat{S}$ and use the conditional normal distribution of $\widehat{X}$.